Overview of Linux device driver
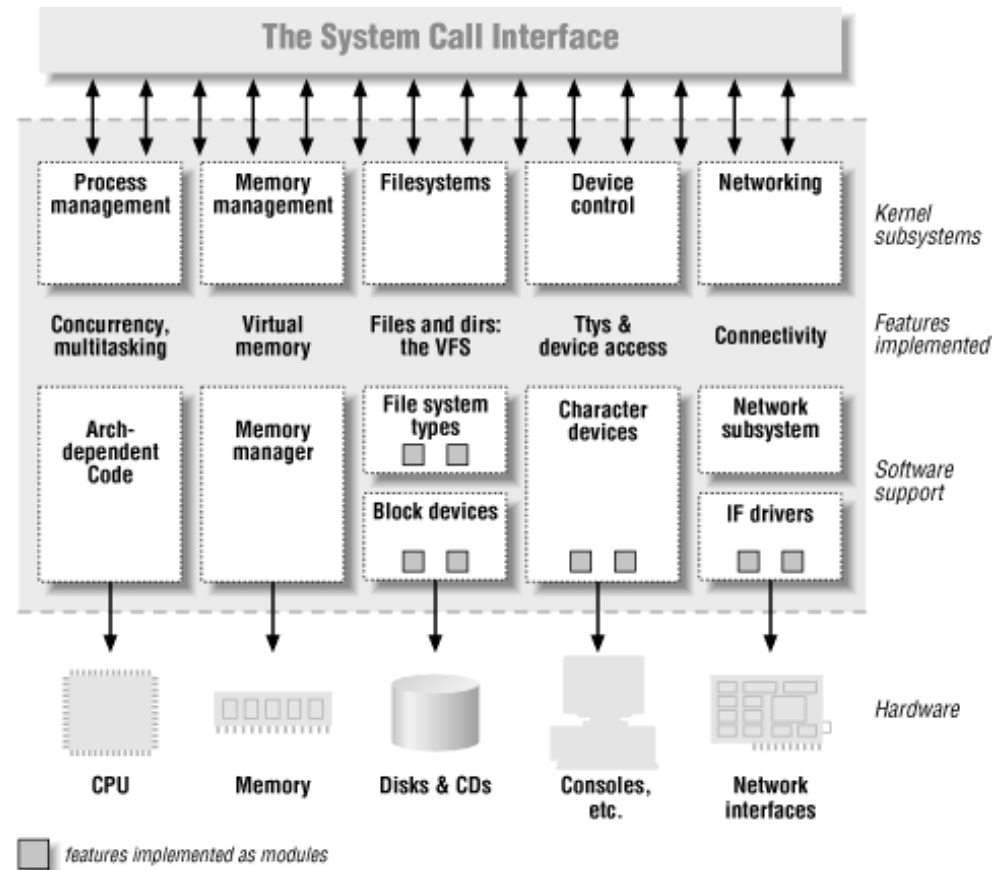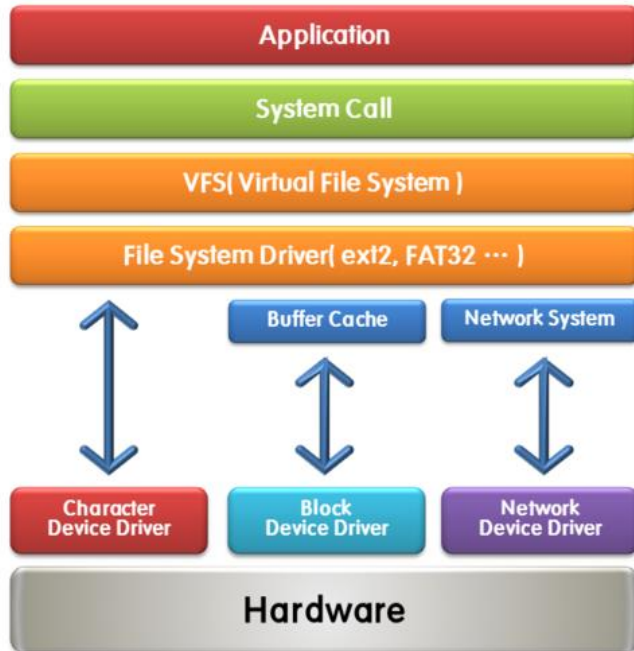
Contents

VC Development Center Vietnam

Tai2.tran

LG
Life's Good

Overview of Linux device driver

Left diagram (layered stack):

- Application
- System Call
- VFS(Virtual File System)
- File System Driver(ext2, FAT32 ···)
  - Buffer Cache
  - Network System
- Character Device Driver
- Block Device Driver
- Network Device Driver
- Hardware

Right diagram:

The System Call Interface

Kernel subsystems	Process management	Memory management	Filesystems	Device control	Networking
Features implemented	Concurrency, multitasking	Virtual memory	Files and dirs: the VFS	Ttys & device access	Connectivity
Software support	Arch-dependent Code	Memory manager	File system types / Block devices	Character devices	Network subsystem / IF drivers
Hardware	CPU	Memory	Disks & CDs	Consoles, etc.	Network interfaces

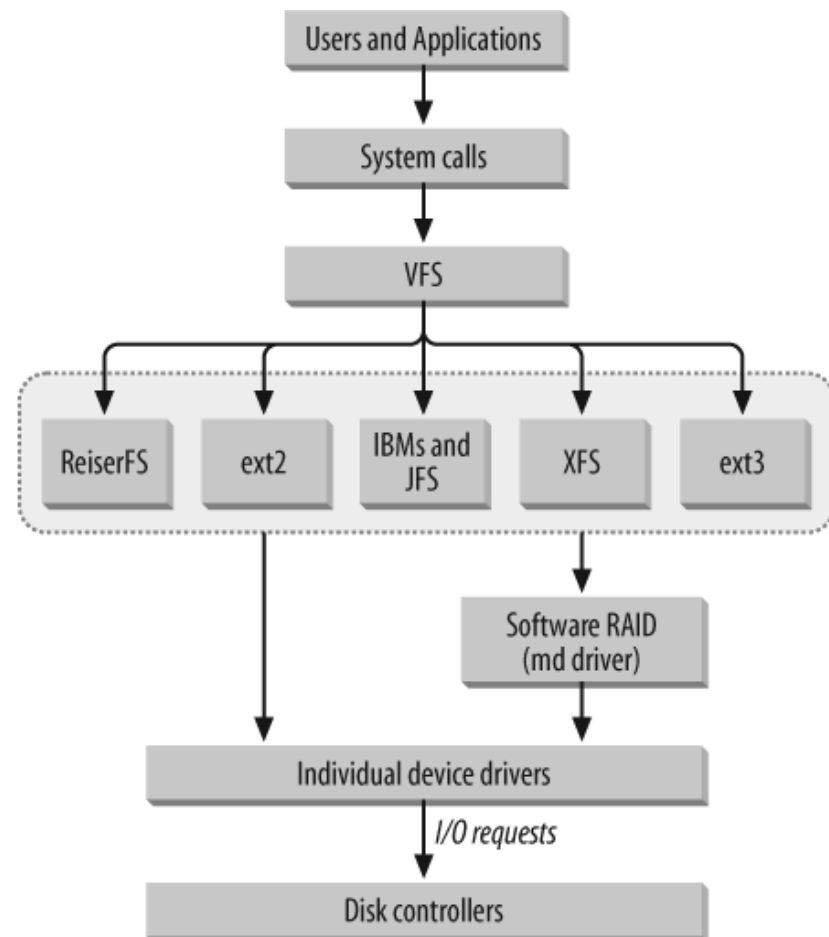☐ features implemented as modules

LG
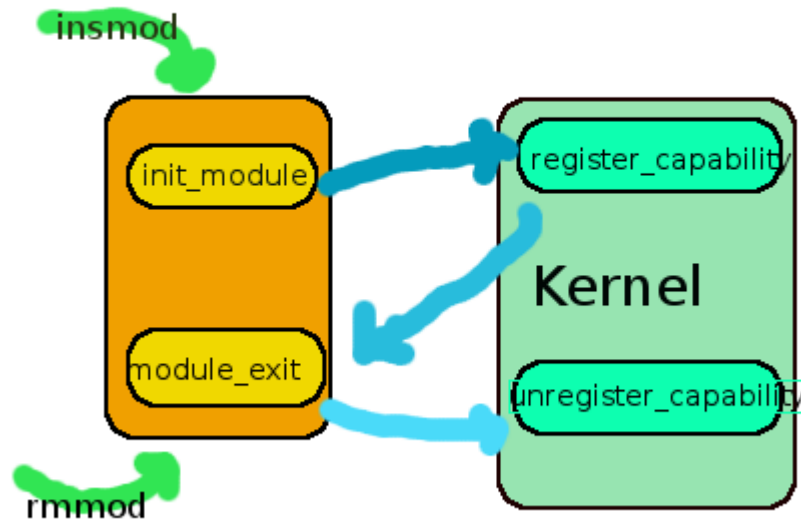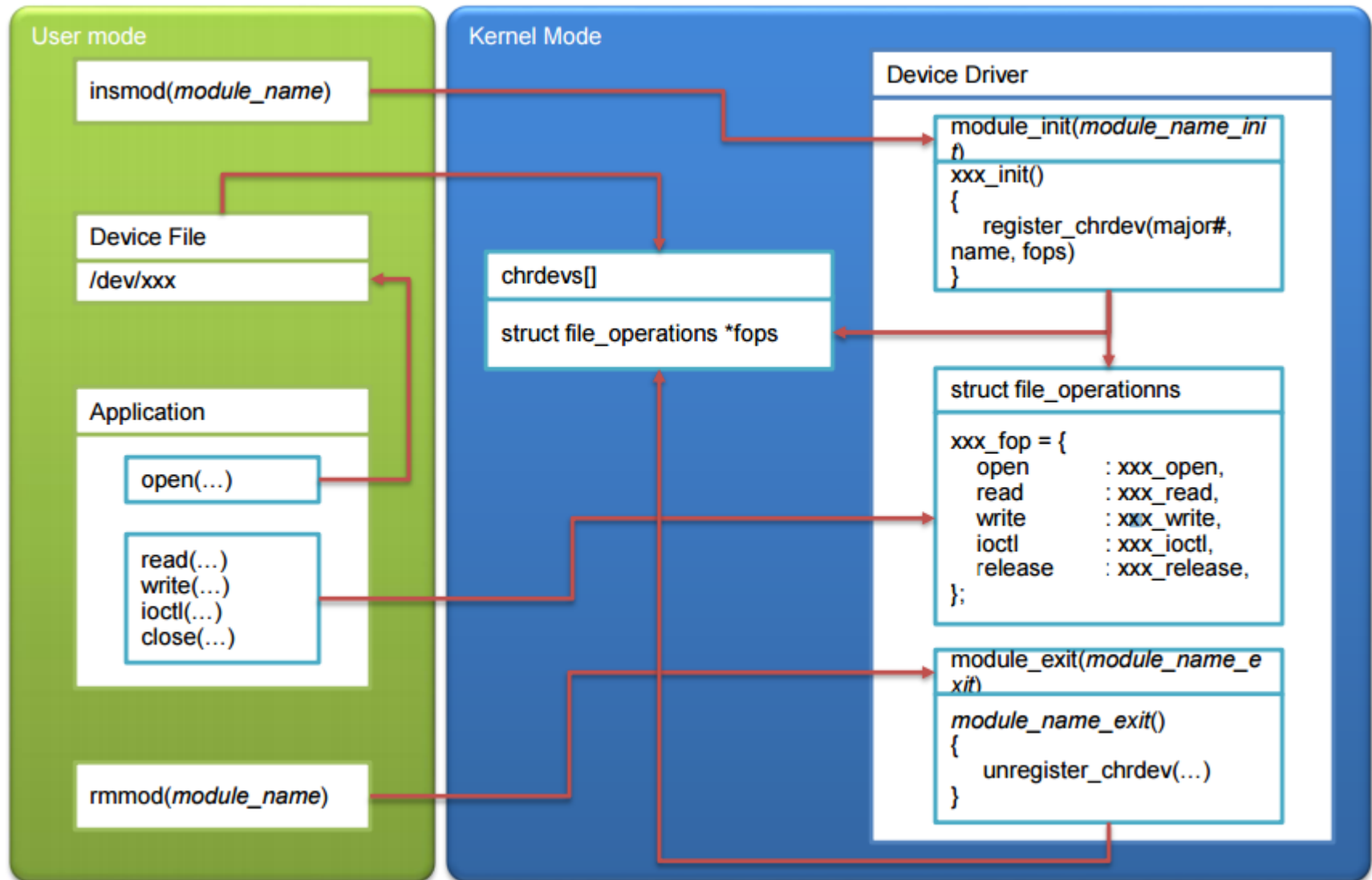Life's Good

Overview of Linux device driver

VFS (Virtual File system)

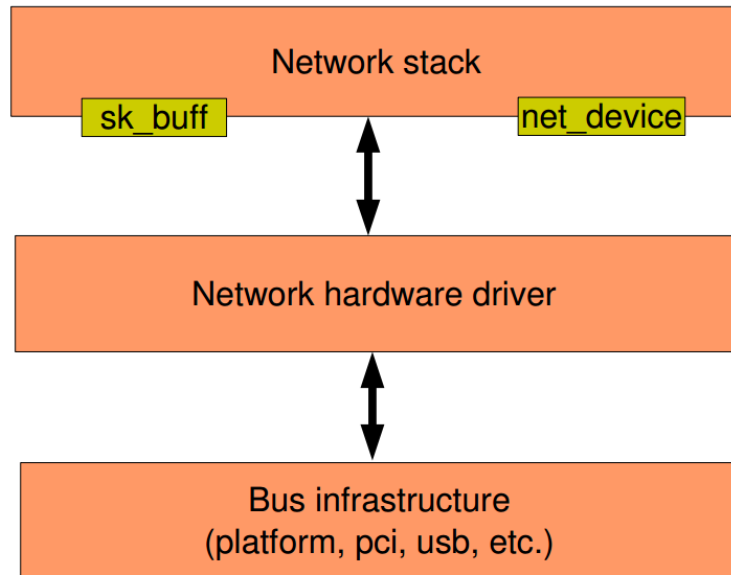- Disk file system
- Special file system (proc, debugfs, sysfs)

Kernel module

Character device driver

Network driver

Network stack

sk_buff   net_device

Network hardware driver

Bus infrastructure
(platform, pci, usb, etc.)

struct file_operations

open()
release()
write()
read()
llseek()
ioctl()
…

struct net_device

open()
stop()
hard_start_xmit()
get_stats()
set_mac_address()
do_ioctl()
…

```
struct net_device_stats
{
        unsigned long      rx_packets;          /* total packets received */
        unsigned long      tx_packets;          /* total packets transmitted */

        unsigned long      rx_bytes;  /* total bytes received */
        unsigned long      tx_bytes;  /* total bytes transmitted */

        unsigned long      rx_errors; /* bad packets received */
        unsigned long      tx_errors; /* packet transmit problems */
        …

};
```

LG
Life's Good

Network driver

❖ struct net_device

- This structure represents a single network interface
- Allocation takes place with alloc_etherdev()
  - The size of private data must be passed as argument. The pointer to these private data can be read in net_device>priv
  - alloc_etherdev() is a specialization of alloc_netdev() for Ethernet interfaces
- Registration with register_netdev()
- Unregistration with unregister_netdev()
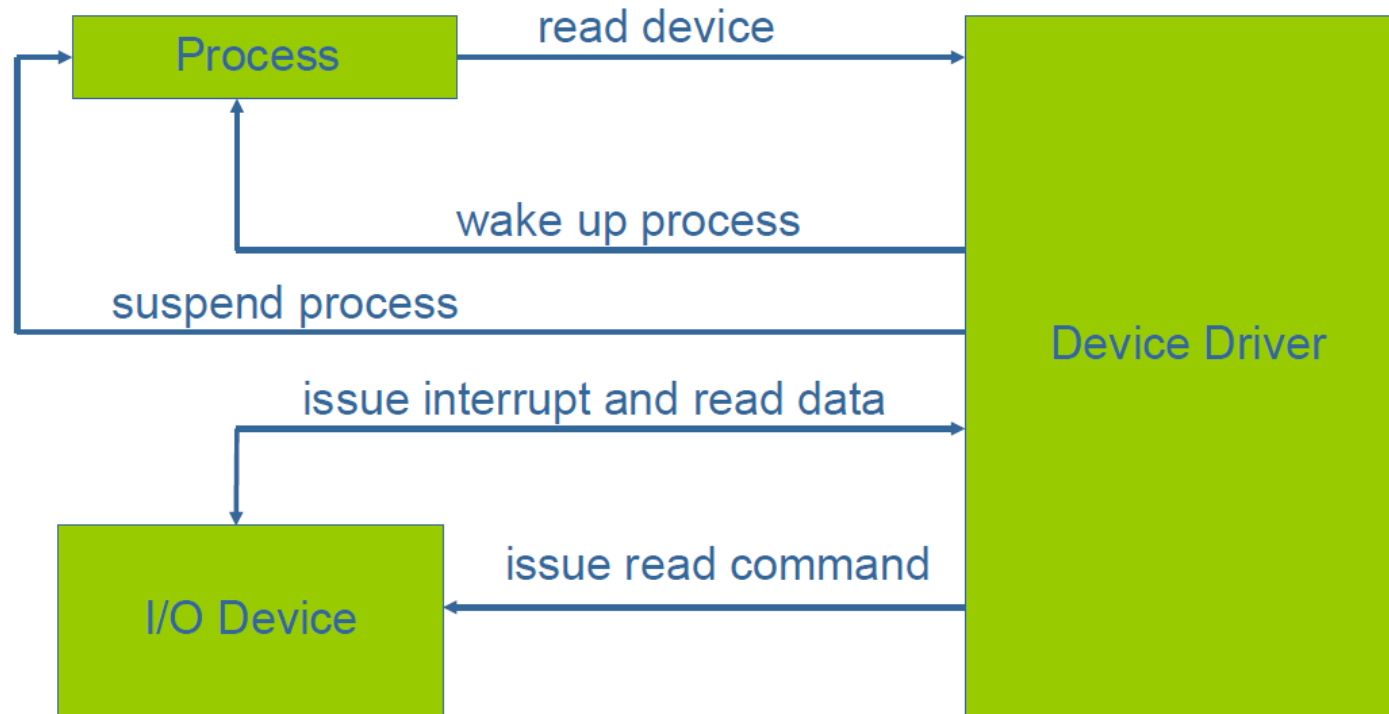- Liberation with free_netdev()

LG
Life's Good

Network driver

❖ struct net_device_ops

- The methods of a network interface. The most important ones:

  - ndo_open(), called when the network interface is up'ed
  - ndo_close(), called when the network interface is down'ed
  - ndo_start_xmit(), to start the transmission of a packet

- And others:

  - ndo_get_stats(), to get statistics
  - ndo_do_ioctl(), to implement device specific operations
  - ndo_set_rx_mode(), to select promiscuous, multicast, etc.
  - ndo_set_mac_address(), to set the MAC address
  - ndo_set_multicast_list(), to set multicast filters

- Set the netdev_ops field in the struct net_device structure to point to the struct net_device_ops structure.

LG
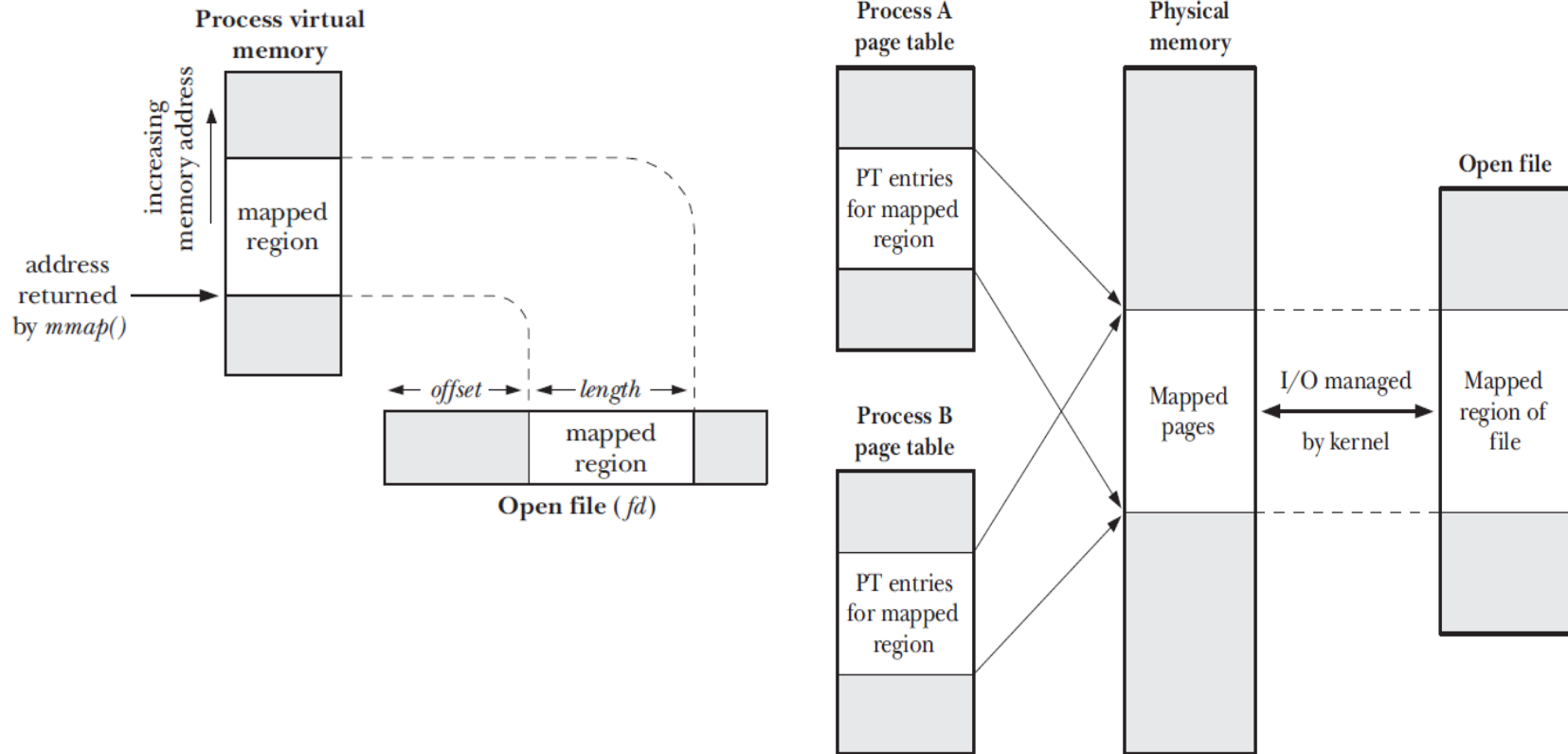Life's Good

IO operation

- Interupt
    - Polling mode (synchronous interrupt)
    - Interrupt mode (asynchronous interrupt)

Communicate between user space and kernel space

- procfs (Done)
    - https://en.wikipedia.org/wiki/Procfs
    - https://github.com/karelzak/util-linux/blob/master/sys-utils/lscpu.c
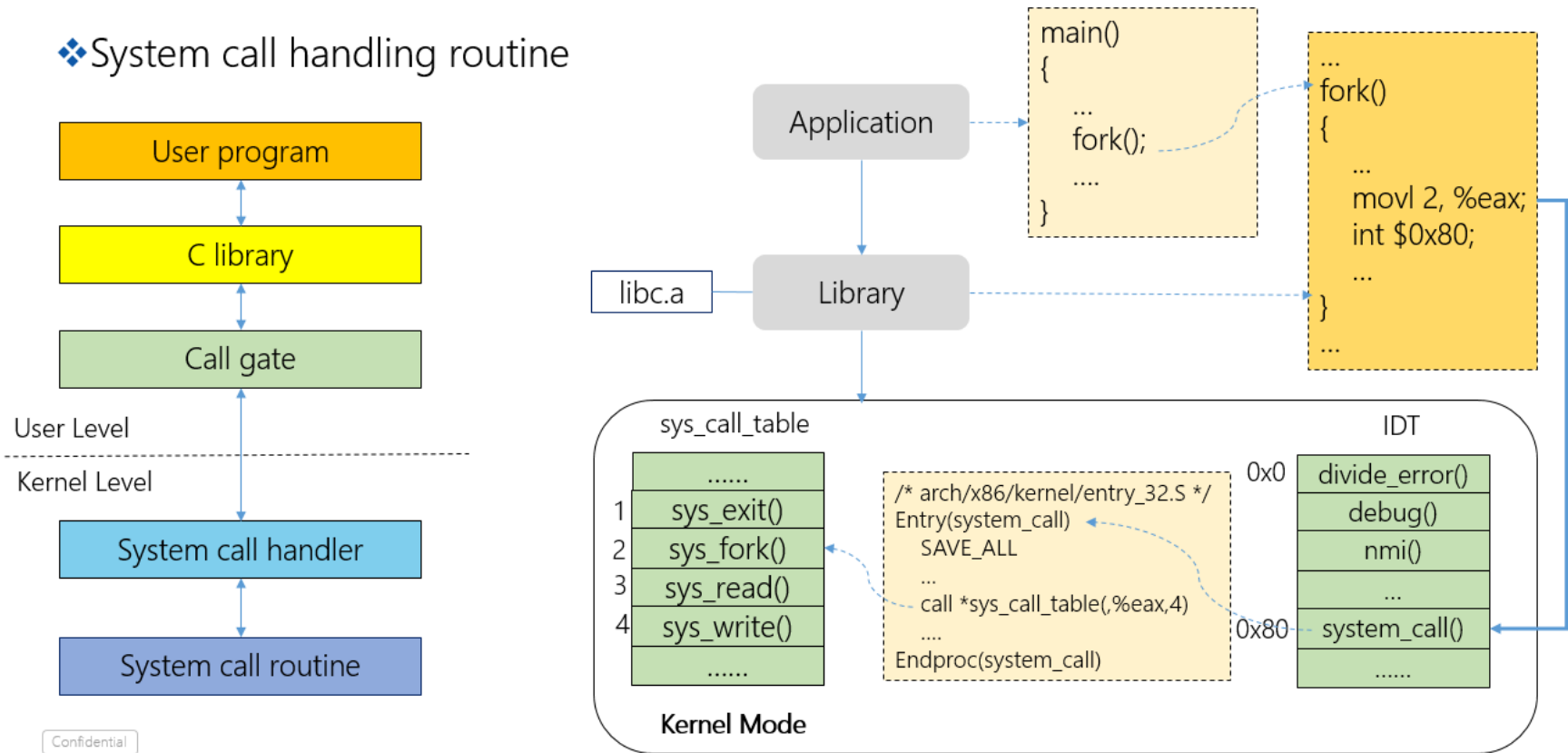- sysfs
- debugfs
    - file, directory is created on fly (based RAM file system) (differ with ext2, etx3)
- character device
- ioctl
- system call
- mmap
- sysctl
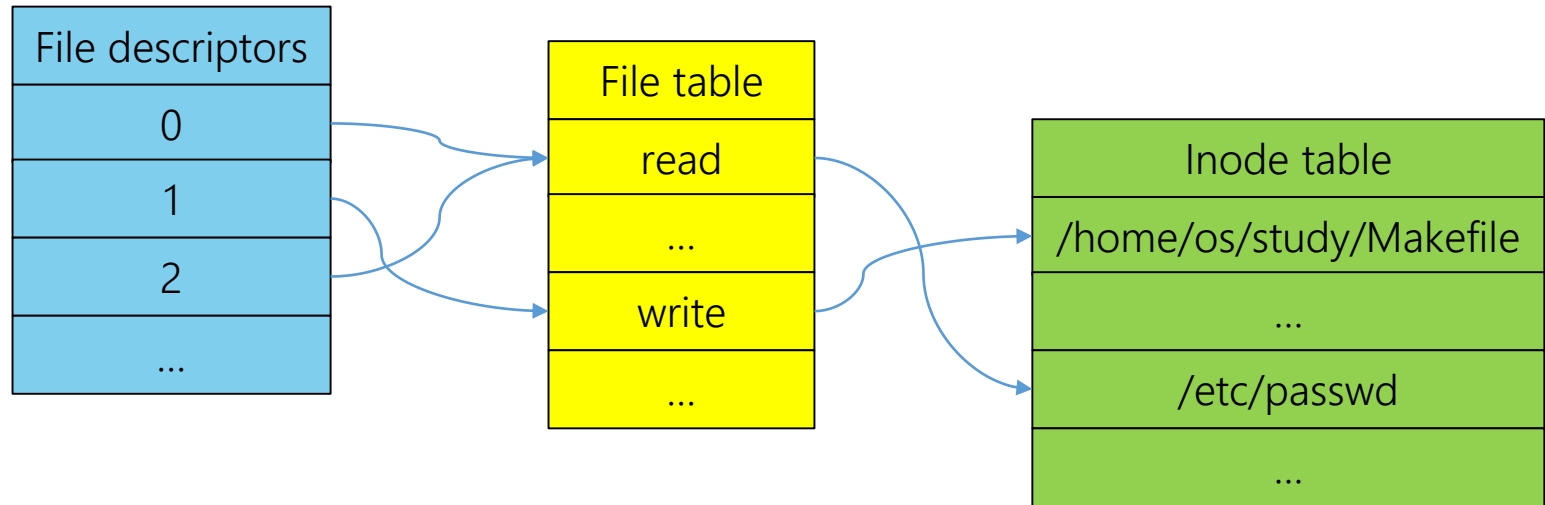- socket based mechanism
    - udp socket
    - netlink socket

Be First, Do It Right, Work Smart

mmap

System call

❖System call handling routine

User program

C library

Call gate

User Level
- -
Kernel Level

System call handler

System call routine

Confidential

Application

```
main()
{
    ...
    fork();
    ....
}
```

libc.a → Library

```
...
fork()
{
    ...
    movl 2, %eax;
    int $0x80;
    ...
}
...
```

sys_call_table

	......
1	sys_exit()
2	sys_fork()
3	sys_read()
4	sys_write()
	......

```
/* arch/x86/kernel/entry_32.S */
Entry(system_call)
    SAVE_ALL
    ...
    call *sys_call_table(,%eax,4)
    ....
Endproc(system_call)
```

IDT

0x0	divide_error()
	debug()
	nmi()
	...
0x80	system_call()
	......

Kernel Mode

LG
Life's Good

File descriptor

- **End of Document**